DIGITGRAPH
P. O. Box 4249
Tucson, Arizona 85717

RSTS-11 SIG DECUS

NOV 1976 RSTS-11 SPECIAL INTEREST GROUP NEWSLETTER VOL 3 NO 5

FROM THE EDITOR

WELL IT IS TIME FOR FALL DECUS. THE SCHEDULE OF RSTS SESSIONS IS THE BEST EVER (SEE PAGE 3), SO COME JOIN US IN LAS VEGAS.

THE NEW EDITION OF RSTS USER PROFILES IS OUT. THANKS TO BILL TABOR FOR THE MANY HOURS OF WORK RESULTING IN A TERRIFIC JOB. AS PROMISED THESE PROFILES WERE ONLY MAILED TO THOSE USERS WHO TOOK THE TIME TO FILL OUT A PROFILE. THE PROFILE FORM IS AT THE END OF THE NEWSLETTER. SEND ONE IN IF YOU WISH TO BE ADDED TO THE SPRING EDITION OF THE USER PROFILES.

SINCE THE LAST NEWSLETTER MY COMPANY HAS INSTALLED THREE 11/34 BASED RSTS SYSTEMS. THE MACHINE LOOKS GOOD AND DEC SHOULD BE COMPLIMENTED ON THE SOLID WAY MACHINES ARE COMING OUT OF THE FACTORY.

I HAVE RECEIVED NO WORD OF THE RELEASE DATE ON V6B, BUT IT STILL SEEMS TO BE SLIPPING AND SLIDING.

SEVERAL RSTS USERS HAVE WRITTEN TO TELL ME THEY HAVEN'T BEEN RECEIVING THE NEWSLETTER. DECUS WAS HAVING A PROBLEM GETTING NAMES ADDED TO THE MAILING LIST AT THE COMPUTER SERVICE THEY ARE RENTING TIME ON. THE BACKLOG HAS BEEN CLEARED, SO HOPEFULLY EVERYONE WILL BE GETTING THE NEWSLETTER NOW. IT WOULD SURE BE NICE IF DEC WOULD LET DECUS GET A MACHINE BIG ENOUGH TO HANDLE THEIR OWN MAILING LIST AND OTHER COMPUTER NEEDS INHOUSE (THEY HAVE A 310 NOW). A 11/34 RSTS MACHINE WOULD BE GOOD FOR THE JOB. INCIDENTALLY, IF YOU WANT TO BE ADDED TO THE MAILING LIST OR HAVE AN ADDRESS CHANGE, PLEASE SEND THEM DIRECTLY TO DECUS. SENDING THEM TO ME JUST ADDS AN EXTRA STEP.

SEVERAL USERS HAVE ASKED ME ABOUT THE CONNECTION BETWEEN THE SIG AND RSTS LOCAL USER GROUPS (LUGS). LUGS ARE SEPARATE DECUS ORGANIZATIONS FOR THE SIG AND ANY DECUS MEMBER CAN PETITION THE DECUS BOARD TO FORM A RSTS LUG IN HIS AREA. THERE WILL BE A SESSION AT DECUS ABOUT RSTS LUGS. I HOPE THOSE INTERESTED WILL ATTEND.

SEE YOU IN VEGAS.....

BOB BRANTON
R LA BAR & ASSOCIATES, INC.
2750 NORTHAVEN ROAD, SUITE 102
DALLAS, TEXAS 75229
(214) 243-3451

Here is a list of the hard working people who are chairing RSTS Subgroups:

A. SUBGROUPS:

COMMERCIAL RSTS USERS:
STU KATZ
COMMERICAL COMPUTER SERVICES
401 FULLERTON PARKWAY
CHICAGO, ILL. 60614
(312) 929-7400

EDITOR OF COMMERCIAL SECTION
MURRY STINSON
SCIENTIFIC PLACEMENT
2700 SOUTH POST OAK ROAD
SUITE 2430
HOUSTON, TX 77056
(713) 621-9811

RSTS V4 USERS:
ROY SWATZELL, JR.
V.A. HOSPITAL
ROOM 2 R 12
BIRMINGHAM, AL 35233
(205) 933-8103 EXT 239

EDUCATIONAL RSTS USERS
HOWARD KIPPLE
NEW MEXICO JUNIOR COLLEGE
HOBBS, NEW MEXICO 88240
(505) 392-6526

LARGE RSTS SYSTEMS
JOAN SALKIN
CROWN ZELLERBACK
1 BUSH STREET
SAN FRANCISCO, CA 94119
(415) 823-5504

B. WORKING GROUPS:

DECUS LIBRARY PROGRAM AND PROGRAMMING STANDARDS DEVELOPMENT
BILL TABOR
FLORIDA COMPUTER, INC.
99 NW 183RD STREET
N. MIAMI, FLORIDA 33169
(305) 652-1710

ARDOTH WILSON
CENTRAL STATE UNIVERSITY
100 N UNIVERSITY
EDMOND, OK 73034
(405) 341-2980 EXT 321

RSTS USER WISH LIST DEVELOPMENT:
BILL SCONCE
UNIVERSITY OF LOUISVILLE HSCC
P.O. BOX 1055
LOUISVILLE, KY 40201
(502) 588-6911

RSTS Sessions at Fall DECUS

The fall DECUS Symposium will be at the MGM Grand Hotel in Las Vegas from Dec. 6 thru Dec. 9. There will be more RSTS sessions than ever before. I have tried to list those sessions directly related to RSTS. In addition there will be many other sessions not listed here of general interest to you.

RSTS DECUS Sessions

Monday December 6th

9:00 - 9:30	Globe Room	RSTS/E Scope Editor
10:15 - 11:45	Mayer Room	RSTS/E Product Panel
1:30 - 2:15	Mayer Room	RSTS Standards Workshop
2:15 - 3:00	Mayer Room	Management of Large RSTS Systems Workshop
3:15 - 4:45	Mayer Room	RMS-11 Tutorial
4:45 - 5:15	Mayer Room	DBMS

Tuesday December 7th

8:30 - 9:30	Mayer Room	RSTS SIG General Session
9:45 - 10:15	Mayer Room	How to start a RSTS LUG
10:15 - 11:45	Mayer Room	RSTS/E run time systems
1:30 - 3:00	Mayer Room	RSTS/E commercial SUB - SIG
3:15 - 4:15	Mayer Room	RSTS/E SIG Wish List
4:15 - 5:15	Mayer Room	RSTS informal discussion Part I
4:15 - 5:15	Orpheum Room	RSTS educational SUB - SIG

Wednesday December 8th

8:30 - 9:30	Mayer Room	DECNET/E (DECNET for RSTS/E)
9:45 - 10:15	Mayer Room	Effective file structures under RSTS
10:15 - 11:45	Mayer Room	RSTS/E workshop system design methods
2:00 - 3:00	Mayer Room	BASIC - PLUS II
3:15 - 4:15	Mayer Room	RSTS Papers Part I
4:15 - 4:45	Mayer Room	Application oriented system - Future consideration
4:45 - 5:45	Mayer Room	RSTS informal discussion Part II

Thursday December 9th

8:30 - 9:30	Mayer Room	CTS-500 Product Panel
8:30 - 10:15	Rialto #2	Educational Papers
9:45 - 11:45	Mayer Room	RSTS Papers Part II: 1. Using batch to control FORTRAN under RSTS. 2. Unified approach to indexed sequencial & Hierachial sequencial access methods. 3. KDSS/TAM Update 4. The interactive systems language
10:45 - 11:15	Rialto #2	Wise Users
1:30 - 2:00	Mayer Room	Workshop: FORTRAN under RSTS
1:30 - 3:00	Rialto #2	Educational papers
2:00 - 3:00	Mayer Room	RSTS/E informal discussion Part III
3:15 - 5:15	Mayer Room	COBOL V3
3:15 - 5:15	Riato #3	Design on Multi-User Business systems under RSTS.

RSTS EDUCATIONAL SUBSIG

The agenda for the Fall DECUS meeting will be:

(1) Update and distribute the Educational RSTS User Directory and discuss the purpose of this SIG.

(2) Discuss "pot-pourri" file, contributors, calls for the file to-date, and direction it should take.

(3) Reports from software users such as: WISE, DECAL, COSAP and discussion of software sought by institutions.

(4) Organization and planning for Spring DECUS meeting and educational concerns of RSTS operating system.

It is possible that this agenda will cramp the one hour allocated; however, since it is scheduled from 4:15 to 5:15 on Tuesday December 7th we could overflow into the evening.

I would like to encourage all educational RSTS users to attend the EDUSIG business meeting scheduled for Wednesday December 8th from 3:15 to 4:15. As an educational user of DEC hardware your ideas, concerns, participation, and leadership are desperately needed in EDUSIG and the RSTS Educational SUBSIG. It should be through the special interest groups that the desires of educational users are met during the DECUS meetings. Your personal contribution will determine the effectiveness of these SIGS.

--- NOTICE ---

Francis Ladd of DEC Business Products Software Documentation Group will be at DECUS to answer any questions users have about CTS Documentation. She would also like to hear any suggestions you have for improving it.

RSTS/E FILE TECHNIQUES
Bit Mapping Availability

The design of the invoicing segment of our warehouse system called for a file structure that normally houses invoices of two to four line items, but at the same time can cope with massive invoices containing hundreds or thousands of line items without wasting space on the small ones.

Lacking a variable-length access method, (or any access method other than IAM at the time) I chose a bit mapped structure. This particular file consists of 16384 blocks of data space. Each block is mapped by one of the bits of four blocks (called the bit map). If the block is available for use, the bit is off, if in use, the bit is on. Each data block can be fielded into six conventional record-IO records of different types to hold the many varieties of line-item detail

The code that controls the bit-map performs only two functions. The first (FNB5%) looks up the next available number, marks its corresponding bit, and returns the number to the caller. The second (FNB3%(I%)) locates and sets off the bit corresponding to the record number I% (supplied by the caller). This effectively returns the block number I% to the pool of free blocks.

The prospective user of these functions must bear in mind that they in no way restrict themselves to mapping disc-blocks. The function merely returns an available number, or conversely, frees a given number for later re-use. The number returned (or freed) can map logical records in a DMS-500 RAM file, or your own record IO file, pairs of contiguous blocks, or something entirely other... the structure is really a dynamic availability queue.

Some coding notes:

1) The bit map itself is presumed open (mode 5%). It is on channel 10% and is fielded once in the housekeeping into 256 words.

2) B6%(15%) is a table of powers of two used to set or clear bits:

```
DIM B6%(15%) \ B6%(I%)=2%**I% FOR I%=0% TO 15%
```

THESE ROUTINES HAVE BEEN QUITE THOUROUGHLY TESTED in production. I use them on two different files, one as described with single blocks, and another with pairs of contiguous blocks.

```
546       GOSUB 25070 !FN146 APPENDED - BIT MAP
25000     DEF FNB5%(I$)!
          ------------------------------------------------------------------------
          INVOICE DETAIL BIT MAP LOOKUP - FN146

25010     FOR B4% = 1% TO 4%\
                    GET #10% RECORD B4%! READ 1ST BIT MAP BLOCK

25015               FOR B8% = 0% TO 255%\
                              B9% = NOT(CVT$%(B8$(B8%)))\
                              GOTO 25030 IF B9%
25020               NEXT B8%
25022     NEXT B4% ! HERE IF BOTH BLOX OF BIT MAP FULL

25025     PRINT 'BIT MAP FULL - CALL FOR HELP'\STOP

          !GOT A WORD WITH AT LEAST ONE ZERO BIT
                    NOW FIND THE BIT

25030     FOR B7% = 0% TO 15%\
                    IF (B9% AND B6%(B7%)) THEN 25040
25035     NEXT B7%

25040     LSET B8$(B8%) = CVT%$ (NOT B9% OR B6%(B7%))

25045     PUT #10% RECORD B4% ! UPDATE BIT MAP

25050     FNB5% = (B8% * 16%) + B7% + 1% + ((B4% - 1%) * 4096%)!
          # WORDS INTO MAP * 16 BLOX / WORD + # BITS INTO WORD
          + ALL BITS IN FIRST BLOCK OF MAP IF LOOKING AT SECOND

25060     FNEND !
          ------------------------------------------------------------------------
          HOUSEKEEP:

25065     DIM B8$(255%)

25070     FIELD #10%, 2% * I% AS D$, 2% AS B8$(I%) FOR I% = 0% TO 255%\
          RETURN !
          ------------------------------------------------------------------------

          BIT MAP RELEASE - FNB3%(B2%)
                    B2% IS RBN OF DETAIL - CALC WHICH BLOCK

25072     DEF FNB3%(B2%) \
                    B2% = B2% - 1% \
                    B4% = B2% / 4096% \
                    B2% = B2% - (B4% * 4096%) \
                    B4% = B4% + 1% !
          !B4% IS BLOCK, B8% THE WORD, B7% THE BIT
25075     GET #10% RECORD B4%\
                    B8% = B2% / 16%\
                    B7% = B2% - (B8% * 16%)\
                    B9% = NOT(CVT$%(B8$(B8%)))\
                    LSET B8$(B8%) = CVT%$(NOT B9% XOR B6%(B7%))
```

```
        !REWRITE BLOCK AND RETURN
25080   PUT #10% RECORD B4%\
        FNB3% = 0%
25085   FNEND!
```

32767 END

SOME APPLICATION NOTES:

My actual invoice file structure consists of a summary record and a linked list of detail blocks. The invoice summary record exists in a separate file and is a simple direct-access arrangement keyed by the invoice number. The entire population of possible invoice numbers are pre-formatted in this file (the summary record is quite small).

Each summary record contains a CVT%$ word in which is placed the first detail-block's number obtained from the bit-map as the invoice is being formed:

```
LSET N9$(18%)=CVT%$(FNB5%(I$))
```

As each block is acquired and filled with detail, its address is placed into the last two bytes of the preceeding block before it is written. Thus, potentially endless chains of detail blocks can be formed for the big invoices, while the little ones are forced to accept only the overhead of one or two unused line items of the six available in a block.

I would like to hear from anyone who uses the routines. I would be most interested in any uses other than mapping records. The user of this package should create a utility that displays the status of the map from day to day, lest one be caught with his bit map full.

Dave Mallery
NATIONWIDE DATA DIALOG, INC
70 James Way
Southampton, PA 18966
215-322-2050

FROM: Jay A. Zittle, Manager of Data Processing/Planning Engineer
Morgan County Rural Electric Association
P. O. Box 738
Fort Morgan, Colorado 80701

DATE: October 15, 1976

SUBJECT: Data Compaction and Extracting Variables from Print Using Statements

I read the letters of Dave Mallery and Paul Grade in the September newsletter with interest because our consumer billing program, among other things, also had some special requirements regarding data compaction and print using variable extraction.

Data compaction: When writing the program, it became apparent early that the storage requirements for the CVTF$ function using 8-byte FP (double precision is a must on our system) would waste too much storage space (for variables such as a consumer's kwh usage which may frequently be larger than the maximum integer value of 32767, but seldom greater than 120,000 kwh per month) and cause unnecessarily large record sizes for each consumer on the billing file. Furthermore, as an added "problem", our processor is an 11/40, so all 8-byte FP manipulations (via software) take more time than hardware FP would. In short, data compaction was imperative to store the entire billing file, but the method outlined by Dave Mallery couldn't be used because of excessive run time. Therefore, the following method was devised which is considerably faster on the 11/40.

```
1Ø          0Ø$ = STRING$ (4%,Ø%)
                '
                '
                '
1ØØ         OPEN "TEST" AS FILE 1%:
          FIELD#1%, 4% AS V$, 4% AS V1$
                '
                '  (Calculate some FP value named K in the range of a
                '   standard 4-byte FP number)

2ØØ         RSET V$ = CVTF$ (K)
                '
                ' (Retrieve some FP value named K1)
                '   previously stored in V1$
3ØØ         K1 = CVT$F (0Ø$+V1$)
                '
                '
                '
32767       END
```

The advantages of this method are speed, simplicity, and automatic decimal tracking (in essence, all the RSET statement in line 2ØØ does is to compress an 8-byte FP value to the corresponding 4-byte FP value since the first 4 bytes are Ø anyway for values from 0.14 x 10^{38} to 1.7 x 10^{38} with seven significant digits; all the statement in 3ØØ does is to restore the 4 zero bytes chopped off by the RSET statement).

Print Using Variable Extraction: The method employed by Paul Grade to extract print using variables is essentially the same as ours with the exception that a Get statement to reset the buffer pointer again uses too much time. To overcome this, I used an error trap as follows:

```
1Ø          T$="": T$=T$+CHR$(X%) FOR X%=Ø% TO 31%: T$=T$+"Ø":
                   T$=T$+CHR$(X%) FOR X%=33% TO 127%
1ØØ         OPEN "TEMP" AS FILE 1%: CLOSE 1%:
            NAME "TEMP" AS "TEMP <62> ":
            OPEN "TEMP" AS FILE 1%: FIELD #1%, 8% AS N$
11Ø         ON ERROR GOTO  8ØØ
                '              (This program prints an 8 digit number with
                '               leading zeroes.)
                '
2ØØ         PUT #1%            (This statement causes an error trap.)
21Ø         PRINT #1%, USING "########", X
22Ø         N1$=XLATE (N$,T$): PRINT N1$
                '
                '
                '
8ØØ         RESUME 21Ø IF ERR = 1Ø% IF ERL=2ØØ%
999         ON ERROR GOTO Ø
```

The advantage to this method is speed because the protection status of the file is stored in core and examined before the system accesses disk, causing a trap to 8ØØ, all at core memory speed, when statement 2ØØ is executed.

Hopefully, these two techniques will be useful to someone. Keep up the good work, Bob. The newsletter is always interesting.

WANTED:

Programmers and Analysts: Experienced in commercial applications and RSTS-11. Working knowledge in COBOL, FORTRAN, DIBOL, RPG. RT considered a plus. Applicant should submit resume and/or call Gary Benson, TELEDATA, INC., 3500 Indian School Rd., N.E.; 505/266-5525.

ALBUQUERQUE, N.M. 87106

Dear Bob,

In reference to Dave Mallery's article of RSTS/E DATA COMPACTION (Double Precision Integers), I have had the requirement of compacting double precision integers. This was accomplished under RSTS/E 4 word math with the following technique:

1) For Storage Use

```
X$ = MID (CVTF$(Z9),5%,4%)
```

2) For Retrieval

```
Z9 = CVT$F (STRING$(4%,0%) + X$)
```

Where Z9=double precision value and X$=4 byte storage

The method above truncates the first 4 characters of the CVTF$, these characters are nulls in double precision integers. In retrieving the value from storage, a string of 4 nulls is added to replace the ones that were intially truncated.

I hope this information can be helpful to Dave Mallery and other RSTS/E Users.

Elmer Ozipko
Software Support Specialist

DIGITAL EQUIPMENT OF CANADA LIMITED
4449 - 99 Street
EDMONTON, Alberta
Canada T6E 5B6

(403)436-6340

This session has been added to the DECUS schedule by Bill Sconce. It will be held at 9:00 Monday Dec. 6 in the Globe room.

RSTS/E SCOPE EDITOR

As Basic-Plus matures, the source text tends to expand: with comments, with multiple-line statements, with longer variable names. Since RSTS' B+ syntax analyzer has no way to selectively update just part of a long statement, much material may have to be retyped to make even a small source-program change.

This paper will outline one promising approach to make updating easier: use of a CRT terminal having the capability to send displayed text back to RSTS. Editing is done by cursor manipulation; no disk is involved; no extra RSTS jobs or run-time systems are required, and the program being edited remains always in core, ready to run.

The implementation to be described has been in use at our installation for 3½ months. We will discuss actual experience, estimated time savings, and attempt a live demonstration.

24-Sep-76

Ms. Cheryl Barber
RSTS11 Library Controller
Digital Equipment Corporation
Maynard, Mass. 01754

Dear Ms. Barber,

I am in complete sympathy with your letter of 19-Aug-76. The points raised with resepect to the current status of the DECUS Library are well understood by me. I also concur that as a user it is my responsibility to do what is necessary to improve the Library material. However, I stand by my comment in the RSTS SIG Newsletter. There are two principle reasons for this.

One reason for wishing to distribute the programs directly is the ability it affords us to know who and where the users of our programs are. we can, I hope, establish with these users the fact that they can rely on programs obtained from us. This can only serve to benfit us in the long run. If we had distributed the material through the DECUS Library, it is doubtful if we would even know who obtained our programs let alone know whether they were satisfied with them.

The second reason is my concern that nothing is really going to be done to make the DECUS Library much more than a repository for programs with a support level of roughly "well it sort of works for us, try it if you like". This would be fine if this type of program could be separated from those in the Library which are of better quality. Indeed I was encouraged by discussion at DECUS in Atlanta that at least one person is going through and 'cleaning up' the Library. I was also encouraged by the suggestion that contributors clearly designate the level of support for their programs when they are contributed to the Library. However, at the Spring DECUS meeting in Boston two years ago the same issues were raised with promises for change made by the DECUS staff. As your letter states, such promises have not been able to be kept, and as a result recent contributions to the Library suffer the same problems with quality control as the old ones. With this being the case, it seems foolish for us to submit our programs to a Library that very few take seriously.

I wish to express my sincerest appologies to anyone at DECUS or any contributors to the Library who may have been offended by my comment. It was not my intent to demean anyone's efforts in this area; rather, for the reasons stated, I simply felt that our contributing to the Library at this time made no sense.

Sincerely,

James A. Krupp
Director of Academic Computation
Middlebury College
Middlebury, Vermont 05753

11

cc: Bob Branton, RSTS SIG

Michael Hall of Lawrence University has found a real "Catch 22" in trying to submit programs particularly CUSPS and a TECO-11 package for RSTS. What follows is a copy of one of his letters to DEC and a copy of the DECUS letter written by the legal department. Along the same lines I have been asked to sign some sort of all emcompassing legal disclaimer for the SIG. I personally hope we can avoid bogging down DECUS and the DECUS library with the legal jargon of lawyers trying to justify their existance.

Bob Branton

Ms. Diane Pekin
RSTS Product Manager
Digital Equipment Corporation
146 Main Street
Maynard, Massachusetts, 01754

Dear Ms. Pekin,

You will perhaps recall our telephone conversation last month, in which I indicated I would be writing to you on several subjects of interest to us at Lawrence (and, I think, to other RSTS sites as well). This is the first of those letters, and deals with the matter of user-generated modifications to Digital CUSPs, RSTS itself, etc.

Over a year ago, I made plans to send several Lawrence-modified CUSPs for V4A-12 and V6A-02 to DECUS, together with some associated programs that were entirely Lawrence code. Since the Lawrence-modified CUSPs contain a good deal of code that we found in the original Digital versions of these CUSPs, and since the Digital versions contain Digital's copyright declaration, I thought it appropriate to check with DECUS on the problems, if any, that submittel might create for Lawrence, DECUS or Digital.

Due to the press of other work, I did not pose my query to DECUS until June of this year. Ferne Halley's response, most kindly given, is attached, "for the record". I mean no criticism of Ferne, DECUS or Digital when I say that I think the policy stated in the letter prepared by Legal, which is the heart of her response, creates at least as many problems as it solves, and is at best in need of considerable clarification. Surely answers are needed to at least the following questions:

1. Precisely who will determine when and if "a substantial portion of the original DECsoftware program ... is ... included"?

2. How can we as contributors to DECUS determine in advance (as we must, if we are to use our resources efficiently) what will and will not be acceptable under the standards in 1.? (In other words, what standards will be applied?)

3. Same as 1. and 2., but as applied to the phrase "adequately separate it" (our modifications) "from the licensed software".

4. As I am sure you know, contributors to DECUS are required to attest that, "I further warrant and represent that I have good and sufficient title and all rights and interest in and to the program to grant such permission to DECUS", the permission being that to reproduce and distribute. (DECUS LIBRARY NEW PROGRAM SUBMISSION FORM, January, 1975) But surely, under

4. (concluded)

law, and given Legal's position as stated in the letter, no contributor can in fact sign any such statement until after the program he or she has submitted has been subject to the review process implied in Legal's letter. Does this mean we are supposed to submit the program with an unsigned form, wait to hear about the Legal review, and then sign the (another?) form? (To do otherwise seems to me to be tantamount to attempted fraud, in that the contributor will have signed his name to a statement he cannot know to be true, which is a felony, at least in Wisconsin.) Surely this Catch 22 must be eliminated as soon as possible, or it could have a stultifying effect on all contributions to DECUS.

5. Is Digital prepared to give written assurance to all contributors to DECUS that it will not take legal action against anyone whose contribution is put in the DECUS library? In other words, will Digital assure us that the review process implied in Legal's letter will be viewed by Digital as both necessary and sufficient to establish Digital's rights (if any) in the contribution?

On the separate but related topic of "unilateral" sharing of CUSPs, I ask that someone in authority at Digital state in writing for the public record just exactly what Digital would do (if anything) if one site sent an "illegal" (in the sense of Legal's letter) program (CUSP or other) to another site. Is Digital prepared to permit this if and only if both sites are licensed for the software involved? Who is to determine, and by what standards, what "licensed for the software involved" will mean? Suppose Site A modifies an RSX CUSP to run under RSTS/E. If Site B wants a copy, and Site A is willing to send same to Site B, will Digital require that Site B have a RSTS/E license, an RSX license, or both? Must Site A obtain "proof" (what kind?) that Site B has the appropriate license(s)? (It is assumed in this example that Site A has both RSTS/E and RSX licences, and that the CUSP is "illegal" in the sense of Legal's letter.) If Digital does oppose such unilateral sharing of "illegal" CUSPs and other programs, how can a site find out if a program it wants to share unilaterally is legal?

To some, all of the above may be nitpicking, but to us it is a most serious matter indeed, and we very much want solid, specific answers. Until we get them, I intend that Lawrence shall contribute no code to DECUS, nor share any unilaterally, that could possibly run afoul of what are to me dangerously vague Digital policies. Your assistance in alleviating our anxiety would be much appreciated.

Yours sincerely,

Michael A. Hall
Director of Computer Services

cc: Ms. Halley, DECUS
Mr. Dicks, Digital, Milwaukee
Mr. Branton, THE RESOURCE SHARER

DIGITAL EQUIPMENT COMPUTER USERS SOCIETY
MAYNARD, MASSACHUSETTS 01754 / TEL. (617) 897-5111 / TWX 710 347-0212 TELEX: 94-8457

Dear _____:

Thank you for your letter of _____ concerning the program you wish to submit to DECUS for distribution.

Your proposal has been considered, and although we agree that it may be of interest to other DECUS members, we will not, however, be able to distribute it. Specifically, it appears that in addition to your modification(s), a substantial portion of the original DEC software program, which is a licensed DEC product, is to be included.

As you are probably aware, DECUS only distributes software among its members which can be freely disseminated, thereby excluding licensed software. DECUS, therefore, cannot and does not accept licensed software for distribution.

Although it may not be practical in this situation, we would, of course, be able to distribute your modification(s) if you can adequately separate it from the licensed software.

We hope that you understand our position and that we may be of service to you in the future.

If you have any questions or if I can be of further assistance, please feel free to contact me.

Very truly yours,

MEMORANDUM

June 22, 1976

Lawrence University
Computer Center
Appleton, Wisconsin, 54911

To: Mr. Hall

From: Mr. Koning

Subject: How to get more out of your PRINT USING statement

In a recent issue of THE RESOURCE SHARER, I saw a comment something like, "Wouldn't it be nice to be able to do a PRINT USING, but get the string it generates in a string variable rather than in a file?". The response to this question, as given in THE RESOURCE SHARER, was that this capability does not exist, and it will not be provided in V6B of RSTS/E.

It turns out that this capability does in fact exist in V6A-02, albeit in an inefficient way. Here's how to go about it:

1. A PRINT USING statement performed to a file that is on a block-oriented device (such as disk) will simply put the string generated into the output buffer and leave it there until the time comes to output an entire block to disk. Therefore, the following program:

```
10 OPEN "TEMP" AS FILE 1%
20 FIELD #1%, 10% AS A$
30 PRINT #1%, USING "#.###",PI;
40 PRINT A$
```

   will print out "3.142".

2. This works fine until you try another PRINT. The second PRINT will put its string behind the one for the first PRINT, which is (presumably) not desired. The problem is therefore how to reset the output pointer that PRINT uses.

3. As it turns out, simply including the statement "50 PUT #1%, RECORD 1%" will do the job. Unfortunately, this means that for every PRINT you do, you will also have to do a disk I/O request (thus incurring overhead) just to reset the output pointer. But, if you really need the capability of saving PRINT USING output in a string variable, it will in fact do the job in V6A-02.

What is really needed is a block structured device like a disk that is much faster. What would be best would be a device that simply throws all data written to it away. We have heard a rumor from Maynard that such a device, called a "Null Device" will be available in V6B. If it were available, the appropriate program would then be:

```
10 OPEN "NL:" AS FILE 1%
20 FIELD #1%, 30% AS A$
30 PRINT #1%,USING "#.###",RND;
40 PRINT A$
50 PUT #1%       !TO RESET THE PRINT POINTER
60 GOTO 30       !SO WE CAN KEEP GOING FOREVER
```

Unfortunately, V6A-02 does not have a null device driver.

To allow us to have this capability at Lawrence under V6A-02, I have written and tested a null device driver; documentation is attached.

If Digital has no objection, perhaps we could distribute the attached to interested parties that take the trouble to contact us.

RSTS/E COMMERCIAL USERS NEWSLETTER

The Commercial Sub-Group of the RSTS/E SIG was formed in Miami in April of 1975 in order to provide the commercial users a vehicle to discuss particular commercial application related problems. We have found in the first three sessions that our attendance and participation has grown geometrically and we look forward to an expanded program in the following year.

The agenda for the Fall 1976 meeting will be as follows:

1) A short organization and committee report segment

2) A presentation by a commercial user utilizing multi-languages within a RSTS/E Shop

   The presentation will include discussion of the interfacing of multiple RSTS/E languages and application

3) Commercial RSTS/E informal swap session (An opportunity for commercial users to make available commercial packages to fellow users.)

4) Informal discussion session regarding commercial problems (i.e. the continuing saga of remote line printers, commercial extensions and commercial problems.)

5) Back to the crap tables!

RSTS MARKETPLACE

DESCRIPTION	CONTACT
1.) A virtual sort subroutine for sorting up to 32,000 records for all files (include virtual). 2.) An easy to use line editor for editing program or ASCII data files. (automatic verification) 3.) A management information system for RSTS. 4.) A jobshop scheduler for the allocation of labor and materials in manufacturing or maintenance. 5.) A complete general ledger accounting system includes balance sheet, statements, journals and schedules such as accounts payable. 6.) An interactive payroll package with 941 & W-2. 7.) A complete automotive agency accounting system for GM, FORD and other dealers. An on-line system. 8.) A package of real estate programs - includes amortization schedules, data retrieval and others. 9.) Financial package for bidding and printing holdings of municipal and government bonds.	Al Cordes Houston Timesharing 9039 Katy Freeway #404 Houston, Texas 77024 713-461-5437
1.) Subroutines to phonetically code last names for storage and retrieval via standard phonetic techniques. 2.) Screen generator programs for the Tektronix 4023 buffered CRT terminal. 3.) Complete letter writing system using paragraph/ letter files to build custom upper/lower case letters from form paragraphs. System allows insertions as well as a user composed paragraph.	Murray Stinson Scientific Placement, Inc. 2700 South Post Oak, #2430 Houston, Texas 77056 713-621-9811
Multiple languages can be used with RSTS/E if you have OMSI's RT-11 emulator. OMSI-RT supports MCRO-11, OMSI APL, and other languages, in addition to DEC FORTRAN IV. 64K words of memory are recommended, but there are several 48K users.	Rusty Whitney OMSI Computing 4015 S. W. Canyon Rd. Portland, Oregon 97221 503-248-5903
1.) A consolidating Accounts Payable, General Ledger, Financial Statement package. 2.) A payroll and Labor Distribution package. 3.) An Accounts Receivable and Sales Distribution package.	Henry Stukenborg Plycom Services P. O. Box 160 Plymouth, Indiana 46563 219-936-2183

RSTS Marketplace (cont)

DESCRIPTION	CONTACT
1.) "DATA BOSS" - A completely conversational generalized DBMS with multiple keys. 2.) COUNTRY CLUB PACKAGE - A complete system including accounts receivable, automatic dues, taxes and gratuities, locker control, cashflow analysis, member spend analysis, optional automatic gratuities calculation by waiter, daily managers report, food and beverage analysis, aging, member progress and activity report, all with complete audit trails. 3.) FINSYS - A C.P.A. WRITE UP PACKAGE - Includes Transaction Registers, General Ledger, Multiple Format Profit and Loss Statements, Balance Sheets with comparisons to budget and last year, Employee Earnings Register, 941's, W-2's, Unemployment Reports, Source & Application of Funds. 4.) PAYROLL - Includes hourly and automatic salary inputs, input listings & proofs, up to six miscellaneous deductions per employee. 5.) BILL OF MATERIALS EXPLOSION PACKAGE - With Production Planning and Inventory Control for Manufacturers. 6.) INVENTORY CONTROL SYSTEM - Including Sales Analysis, Reorder points, etc., for wholesalers.	John H. Wright Florida Computer Inc. 99 N. W. 183 Street North Miami, Florida 33169 305-652-1710
1.) GAAP - A General Accounting Application Package. A complete, interactive, on-line, multi-user system supporting the areas of general ledger, accounts receivable, accounts payable, financial reports, and payroll. 2.) DBM - A general purpose Data Base Management Facility. User defined variable length records, variable length fields, and records with repeating groups. 3.) General Utility Package - Includes program for file formatting and listing, checking for possible file corruption, reviewing and modifying the contents of disk blocks, restructuring a data base for increase system efficiency, etc. 4.) An on-line, interactive school administration package including advanced scheduling and grade/attendance reporting capabilities. 5.) A commercial timesharing accounting system - Accumulates raw use data (connect hours, CPU time, storage) by system account number and converts to automatic billing.	Peter Melvin Intelligent System Corp. 460 Tottenpond Rd. Waltham, Massachusetts 02154
GISMO is an index sequential access method for users of DEC's RSTS/E (CTS/E) operating system *Random or sequential add, delete, and update *No searching of overflow chains *Multiple user support *Generic keys	Leonard Farano Gambit Management Strategies 1440 Broadway New York, New York 10018

RSTS Marketplace (cont)

DESCRIPTION	CONTACT
"FIPS" - Financial Information Processing System. An easy to use computerized accounting system designed by accountants for accountants. Includes general ledger, payroll, payables, cost accounting, financial statements, etc.	Michael M. Niernberg Environmental Developers, Inc. 13693 East Iliff Avenue Denver, Colorado 80232 303-755-3377
"PRODUCT 3" - Data Base Management System. Featuring Relative, Keyed Sequential (ISAM), Multi-list chained Sequential access methods, plus multi-user on-line data base management. Available for RSTS/E, RSX11-D, and RSX11-M.	ELS Systems Engineering 2800 Mayfield Road, #203 Cleveland Heights, Ohio 44118
1.) TAM, a format-language-driven terminal-handling front-end software module for transaction-processing application under RSTS/E. 2.) KDSS, a key-to-disk data entry subsystem under RSTS/E. (In both TAM and KDSS, the performance-critical portions are implemented in PDP-11 assembly language and installed under RSTS/E as run-time systems.) 3.) FSORT, a functional equivalent to the SQWIK and MQWIK modules of the RSTS-11 SORT (except that FSORT handles 64K records, rather than 32K), implemented in PDP-11 assembly language as a run-time system under RSTS/E to run anywhere from 8 to 30 times as fast as its BASIC-PLUS counterparts. 4.) LABX, a software package supporting the DEC LPS-11 Laboratory Peripheral System hardware under RSTS/E. 5.) INTCOM, a set of BASIC-PLUS programs allowing two RSTS/E systems to communicate, including ASCII file transfer, via telephone line up to 300 baud.	Tom Evans Evans Griffiths & Hart, Inc. 1844 Massachusetts Ave. Lexington, Massachusetts 02173
1.) A sophisticated flexible and efficient line and text editor. 2.) Maps - financial planning and reporting language.	Ross Systems 1900 Embracadero Rd. Suite 108 Palo Alto, California 94303 415-326-1054
RSTS QED text editor - This QED is an adaptation of the standard QED type editor, written in basic for RSTS, and is a fast, extremely versatile program. Some of the features available in this editor are: *Simple command forms. *Simple access to any line in the file. *Versatile and powerful character-string matching capabilities. *Easy file merging and file splitting.	Marc L. Cohen Four-Site Computer Corp. 39 Rosemont Terrace West Orange, New Jersey 07052 201-736-9078

RSTS Marketplace (cont)

DESCRIPTION

FN1976 - A Sequential Access Method
-for multiple (up to 12) TYPE-1 HEADER FILES
-input, output, add-on and update functions
-any combination of logical record lengths, automatic fielding
-package includes MERG76, a generalized 2 to 10-way merge to disc or tape

CONTACT

Dave Mallery
Nationwide Data Dialog, Inc.
70 James Way
Southampton, Pennsylvania 18966
215-322-2050

1.) Complete Multi-journal System with A/R,A/P,G/L, etc.
2.) Complete Payroll/Personnel System.
3.) Retail Inventory System.
4.) Industrial Inventory System
5.) Mail List System.

Mikell D. Becker
TeleData, Inc,
3500 Indian School Rd., NE
Albuquerque, NM 87106
(505) 266-5525

RSTS/E WORKSHOP SYSTEMS DESIGN METHODS

This RSTS/E Workshop will call upon four (4) panel members to present their different viewpoints in a discussion of the methods and procedures of designing a RSTS/E application.

Panel members and their representations are as follows:

Mr. Robert Branton - Software House
Mr. Philip Seifert - In-House Installation
Mr. Bill Sconce - Educational Field
Mr. Gerald Hornik - Digital Representative

The panel members will define the system design methods and procedures as they apply to their applications. They will provide the workshop with examples of how they proceed in implementing their designs.

Time will be allocated to an informal group discussion session. This will be an informal panel and participation of all commercial users is encouraged.

Looking forward to seeing you in Vegas.

It's time for another instalment in the continuing saga of our RSTS "wishlist". As you can see, the list is growing. In fact, it's become necessary to do some categorizing and indexing to keep things in control. (The index is at the end; looking at it first may make reading the rest easier).

Most of the material for this edition came from notes and discussions at Atlanta. Although some of it is therefore several months old, and for a while we were thinking version 6B would be past tense by now, most of the items still seem fairly timely. As usual, no attribution of "officialness" can be made to any of the opinions offered here, which are only a user's editorializing on the best information which we've been able to gather from DEC.

There are two types of item in what follows: open questions (real "wishlist" entries), and "resolved isssues". Open questions are indicated by a "?", and are offered with little boxes for your ranking on the questionnaire. "Resolved issues" are indicated by a "!", but candor dictates that we recognize that even these can be considered as active questions if there's enough user response. Feel free to offer complaints and suggestions wherever you feel they're warranted.

- -

The response from the last questionnaire was very gratifying, particularly since so many took the time to offer written suggestions of their own. All of them have been packed off to Maynard, and for the next published "wishlist", we'll find some way of incorporating the CUSP suggestions into the mass. We're also going to need some way of keeping tabs on old items, and on problems resolved, or DECUS is going to end up bankrupted by the postage bill.

See you in Las Vegas !

Bill Sconce
University of Louisville HSCC
PO Box 1055
Louisville, KY 40201

(502) 588-6911

1. Field Performance (!)

Question. Let's have representatives of Field Sales and Field Service available on a panel at the next DECUS.

Discussion. DEC has done very well by RSTS in providing to DECUS the time of those most responsible for the development of the product, and much useful dialogue has resulted from panel discussions. Indeed, much credit is due these people, not only for taking the inevitable flak, but for establishing an atmosphere that says "we hear you". Further, it appears that things DO happen because of exposure to the community in which RSTS must make a living. However, it has more than once seemed that some issues of sales support and of field service generated a crossed-up dialogue, since those DEC representatives present were in charge of other areas. We've had excellent access to the software developers, the product managers, the documentation supervisors, the software communications people. But anyone who's attended DECUS has seen them having to field questions that properly should be answered by members of the marketing and field organizations. Perhaps inviting them to the next DECUS will help to fill a gap on the panels.

2. Maintenance Hotline (?)

Question. How about a phone number in Maynard which a user could call when in trouble with software?

Discussion. This suggestion was made last fall at DECUS Los Angeles. DEC's reaction was not unfavorable at that time, and evidently the matter has been under consideration since; no action has yet been taken, however. The thought was that if circumstances warranted, a user willing to pay for a long distance call could get an immediate response to a problem that had his system down. The feeling we got in Atlanta included some misgivings. There would be the problem of creating a staff to insure that there would always be someone to answer the phone. Then, too, it could be a difficult thing to control: phone interruptions could adversely affect normal SPR response, to say nothing of time taken from development. According to DEC's figures, SPR turnaround has been improving, and this is where most effort is going to continue to go. There seemed indeed to be less argument about treatment of SPR's this time.

3. Upgrades Without Maintenance (!)

Question. One of the main reasons for taking out a Software Maintenance subscription is to be first on the list when a new release of RSTS/E comes out. Is it possible for someone who has let his subscripion service lapse to get back up to date?

Discussion. Yes. There is provision for an installation to pay a one-time charge to receive a copy of the latest release. Thus, a subscription-less shop now running 5A could presently obtain 6A for $800. Note, however, that this form of upgrade doesn't include the next year's subscription service; the

latter, if desired, would have to be purchased on its own.

4. Software Licensing for a Monthly Charge (?)

Question. All DEC software is currently available only for a one-time license charge. (One might typically add another yearly figure for subscription service). More than one user has suggested that at least some packages would find easier market entry if "lease" arrangements were available.

Discussion. Those of us working for government agencies will attest to the relative hassle of capital vs operating expenses. For any user, though, justification of a monthly charge is easier and more immediate. This may be the case even if over a five-year product life (for instance) the total lease payments added up to more than the one-time license. Other interesting alternatives might include one-, three-, and five-year lease/purchase arrangements.

5. Microfiche Prices (?)

Question. Microfiche listings seem like a good form of documentation for those needing access to internals. Could the pricing structure be rationalized to make microfiche a more reasonable investment?

Discussion. A set of RSTS/E microfiche costs $1200 - and this figure evidently applies to whatever version of RSTS you might be after. (In any case, you only get one issue; the fiche arrangement is not a continuing subscription). Mentioned in Atlanta was the hope that either a continuing "license" arrangement might be offered, or a discount on fiche for back releases, or both.

6. Programming Tips (?)

Question. There seems to be a never-ending thirst for cleaner ways to do things, ways to save execution time, etc.

Discussion. There have been several treatments of this subject, unfortunately in scattered sources. They range from Software Service Notes, to the RSTS/E Programming Manual, to a handy summary about to be published for the Version 4 sub-SIG. It would be nice if there could be a compendium (perhaps in the Prog manual) where a majority of this information could be found, and be kept up to date. A footnote: Mr. Tadahiko Ohta, of Sumitomo Electric Industries, delivered an applications paper (B15.3) at Atlanta on a remarkable study of RSTS/E efficiency. It deserves to be called THE definitive study at this point; SIG members should have by now received copies from Maynard, kindly provided by the RSTS product staff.

7. Machine-Readable Documentation (!)

Question. Asked in LA, and again in Atlanta: we'd buy RUNOFF-format documentation if it were available, especially the B+ Manual and User's Guide.

Discussion. There are several reasons to want to be able to generate parts of the RSTS/E documentation locally. It's difficult to stock just the right number of general-use documents, particularly in an educational environment. An extra copy could be only a few minutes away on a printer. Customized manuals, reflecting local system features, could be built around the standard DEC material. There are some obvious caveats: no artwork would come on a RUNOFF file, for instance. Also, the version of RUNOFF DEC uses to prepare our manuals is an advanced (RTS type) program, and it uses somewhat different commands from the B+ program so far distributed to the field. This suggestion has now been considered (and approved) in the necessary areas within DEC, and now awaits a decision on pricing strategy.

8. Manual Update Subscriptions (?)

Question. A shop having many copies of manuals out in its user community would benefit from a subscription service.

Discussion. This would mean that Software Communications would have to keep track of who out there had how many of which manuals, and then send the appropriate number of each update. This can obviously be done, but no one knows how much the service would cost, or indeed, how many users would be willing to pay for the service.

9. Page-Insertable Documentation Updates (!)

Question. Updates to manuals (via Software Dispatch or otherwise) need to be distributed in better form.

Discussion. Future updates will be in the form of complete pages which can be simply copied and filed in their manuals. Replacement dates, marginal update bars, and "bullets" for deletions will be used.

10. Program Logic Manual (!)

Question. There's evidently going to be an internals manual prepared with version 6B. Any illumination?

Discussion. Such a manual is indeed in the works, and will describe such things as monitor structure, job device tables (which are being changed for 6B), and so on. It will be ready in the 6B time frame, but no decision has yet been reached on whether to ship it to all users, charge for it separately, or to restrict its distribution. Also, an important caution: publication of table structure, etc, is not a "freeze"; both structure and the manual could certainly in subsequent releases. So if some of us write code that depends on this new documentation, we'll at the least become permanent subscribers!

11. RTS Documentation (?)

Question. RSTS now supports multiple run-time systems (also called object-time systems, RTS, or OTS). Basic-Plus, Fortran, and Cobol each have one; new languages will have theirs; and SORT-11 executes using pure assembly-language code in an RTS. If used heavily enough, any program can be enhanced by implementation in pure code. The TECO editor works this way; so do several OEM packages -- but they were written by people having more than the average insight into the workings of RSTS, to say the least. How about publishing specs on how an RTS must be constructed?

Discussion. There are no current plans to make these specifications widely available. Several things should be considered. Publishing any documentation involves non-trivial expenses, which are difficult to spread equitably if only a few installations will use it. Publication would not be a "freeze" or guarantee that the specs could not change in later releases, forcing updates of any user-written RTS's. Finally, customers desiring to use their own RTS facilities would probably have to waive DEC support for that part of their systems.

12. Queue Protocol Documentation (?)

Question. Documentation on the functional design and record layouts used by QUEMAN?

Discussion. More than one user has expressed an interest in building local packages to operate within the standard queue structure. This would require knowing the functions performed by the queue manager, message formats, etc.

13. Bootable Backup (?)

Question. With current software, it's possible to create backup copies of SY: that can be booted to do a stand-alone restore in case some horrible event befalls the system disk. Using $BACKUP, the receiving disk can itself contain a minimum monitor structure. Using ROLLIN, the receiving disk is indeed a complete copy, including bootstrap code. A magtape made with ROLLIN can be booted to start a restore process, too. In any of these cases, it's possible to begin rebuilding a system given only the backup medium, a CPU, and a working drive for SY:. But things are changing. Block-copy techniques will not work on forthcoming disk devices: this means ROLLIN (and PRESRV, too) are on the way out. Performance considerations dictate that the new $BACKUP use a fast output structure, so its copy won't be file-structured.

Discussion. All this seems to indicate essentially one of two things for a system-disk disaster: extra copies of SY:, or a change in the new $BACKUP. Since the restore part of $BACKUP will have to run under a RSTS monitor, it will be necessary to have that monitor available on an extra disk. (This raises the question of keeping the spare monitor up to date with patches, keeping records, etc). Alternatively, it would do to have $BACKUP write a bootable mini-monitor on the backup medium - just enough to somehow get the restore process started.

14. ROLLIN in v6B (!)

Question. Will ROLLIN (the stand-alone image copy program) be available in version 6B? As disk technology advances, ever-increasing recording densities will begin to force the use of media containing bad blocks. Our current reliance on block-for-block copies will not work in this new world; to be meaningful, a copy program will have to know something about file structure.

Discussion. For this (and other) reasons, ROLLIN is out of the mainstream. It will still be distributed in the version 6B kit, but it will no longer be a supported product.

15. Immediate Mode Lockout (?)

Question. An ability to deny a certain job access to immediate-mode statement execution.

Discussion. The thought here is to provide for those jobs which are to run-only production programs, and which are likely to be run by naive users.

16. ROLLOUT-ROLLIN (?)

Question. Has this ever happened to you? You're debugging a program; it has STOP's sticking out all over it, you've modified several variables, and so on. The next thing you need is to look at some part of the input file, but naturally don't want to lose the status of your current job. If you work in the computer center, you might just go to another terminal, but even that is an interruption. This facility would let you save your porcupined program, current status and all (ROLLOUT), run PIP or whatever, then restore the debugging job exactly where it was (ROLLIN).

Discussion. This item was in the January Channel 13, but several readers found the writeup obscure. From the comments in Atlanta, it sounds like users feel it would be a useful facility.

17. Accounting Distinctions (?)

Question. Some of us need to be able to charge for various RSTS services with a finer resolution. For instance, it would be helpful to distinguish between dial-up and local lines; or between various types of device time.

Discussion. RSTS currently supplies lump-sum figures for all connect time and for all device time. In some environments, it's desirable to set up different charge rates for local vs modem lines, or for printer vs card reader vs RP03 device time. Charge rates are one of the system manager's tools for encouraging equitable use of available resources by the user community. Currently, though, differential data on various devices must be captured by local extensions to RSTS.

18. Accounting Extensions (?)

Question. More facilities for system control and accounting would be most helpful, particularly in an environment where computer services are provided at a charge. RSTS could be improved by providing some kind of hooks to point to extended control information for each PPN. Typical needs might be for PPN name, associated billing (receivables) account number, and LOGIN-related data, such as lines authorized, legal times for access, etc.

Discussion. The closest thing in present RSTS is $ACCT.SYS, but it's not suitable for on-line use, being a sequential ASCII file. (In fact, the case could be made that the "S" function in $REACT would be better driven by a file created directly from an old MFD by an extraction program). Although it's hard to see how any fixed facility could be built which would serve all users' needs, perhaps a RSTS-maintained MFD pointer could index to some bucket in a data file. This could then be formatted according to local requirements. The pointer itself could be gotten by a SYS call, and user extension looked for on-line use by SYSTAT, LOGIN, MONEY, etc as needed.

19. Century-Date Function (?)

Question. The existing RSTS/E date-handling doesn't do enough to be fully useful in the applications world.

Discussion. RSTS' date storage technique (and the DATE$ function) were designed to take care of the limited calendar necessary to run an operating system. This means keeping track of the current date, recording dates in file labels, etc. It does not do enough to handle real-world applications, where it's necessary to work with dates earlier than 1970, compute differences between dates, etc. A better facility would be a pair of functions (both from and to internal representation) that would relate a calendar date to a simple number, such as the number of days since 1 January 1850. (That's right: 1850; there are still people in computer files having nineteenth-century birthdates!)

20. LENGTH Function and Command (?)

Question. It would be helpful if non-privileged programs had access to their length, possibly via a Basic-Plus statement similar to the monitor command LENGTH. Also, LENGTH itself would be more useful if it returned an exact figure rather than one rounded to the nearest 1K.

21. CCL in Other Than [1,2] (?)

Question. CCL's must currently be supported by programs stored in [1,2]. Wouldn't it be possible to implement CCL's through any user-specifiable PPN?

Discussion. A couple of benefits could accrue by moving CCL programs out. For one, the philosophy behind concise commands - rapid execution of oft-used functions - suffers because the supporting programs must be found in what's usually a large UFD. (For user functions, those programs are often down at the

end of the UFD, too). For another, some users would like to keep maintenance and documentation by providing only DEC standard programs in [1,2] or "$". It might seem preferable to install local extensions in another PPN for ease of management.

22. RUN PROGNM/xxx (?)

Question. A facility to pass operating parameters to a program via the RUN command could be helpful.

Discussion. A possible syntax could be "RUN PROGNM/xxxx.." where the xxxx.. is information for RSTS/E to stuff into core common for PROGNM to use. A facility of this type would in essence extend CCL-type commands to any program, but wouldn't require extension to the CCL table itself.

23. ASSIGN by Logical Name (?)

Question. RSTS now provides for logical names for devices, but it's still necessary to wire into programs knowledge of object-time file names. How about logical ASSIGN capability for file names as well as devices?

Discussion. This is a typical facility in batch-processing operating systems. It's possible to let a RSTS program read file names from a control file, or course, but the standardization of operating system support is lacking. Also, the interface with BATCH isn't very natural.

24. RUN at Another KB (?)

Question. "I write many of my programs using an editting CRT. When I'm done, I can get a program listing on paper by typing SAVE KB2: where KB2: is a DECwriter not in use. But if I want to execute the program on a hard-copy device, I must log onto it first. I would like to see a command like RUN KB2: which would allow me to see the execution of the program at an alternate ternal."

25. Security of Old Files (?)

Question. RSTS/E has a couple of gaps in its security provisions for disk files. The problem has to do with what happens to old blocks after a file deletion de-allocates them: their information is still on the disk. A nosy user can, via a form of roulette, read them - he opens a new file, pre-extends it to grab up a lot of blocks, and proceeds to browse through their previous contents.

Discussion. This is the problem that the $PIPEXT erase option was created to handle: with it one can clear out a file's blocks before de-allocating them (before KILL'ing the file). The security gaps have to do with program code. 1) -medium- Few users probably think to clear out their TEMPxx.TMP files before logging out. Those old blocks contain source code, and can be read. 2) -severe- When you do a REPLACE, the old version is KILL'ed on the spot, and its former blocks cast to the winds. There's no

chance to PIPEXT clear the file before REPLACE. The only out seems to be... a) SAVE the program under a new name. b) PIPEXT clear the old program. c) NAME newname AS oldname. It would be a big help if a user could say PURGE to RSTS/E, and then rest assured that every block de-allocated later in the job would be first be cleared. The overhead of clearing wouldn't really be an issue, since it would only be invoked at the user's request. A footnote: COMPILE writes a core image on disk, so unless you COMPILE immdiately after OLD, your .BAC files may have snatches of readable code (buffers, etc) spread thrugh them. (Also, a .BAC COMPILE'ed after some RUNning or debugging may well be larger than the code alone would dictate immediately after an OLD).

26. FIP Status Storable Through Chaining (!)

Question. When one does a "CHAIN", all files are effectively closed. The subsequent program must OPEN all files it wished to use, and possibly read to an initial position as well. Much of this overhead could be avoided when the chaining program already had the file structure set up -- IF it were possible to pass the File Processor status through the CHAIN.

Discussion. This looks possible - but only reasonably in Basic-Plus-Two. Support for it in this form will probably be included in version 6B.

27. Read-Only; Read-Regardless (!)

Question. There are times when a program would like to open a file read-only to guard against doing inadvertent damage. Further, one would sometimes like to open a file for read-only even though another job may be in the process of updating it. (It's presumed that the job doing the reading knows how to take its chances about data changing on the fly).

Discussion. These processing modes, which might be called Read-Only and Read-Regardless, are planned to be available in version 6B.

28. Disk Space Allocation (?)

Question. RSTS currently provides no direct mechanism by which one can specify the actual disk addresses a file is to occupy. (Indeed, one can force even contiguity only by setting a monster CLUSTERSIZE, and the location is still unpredictable).

Discussion. This was considered as a possibility for version 6B, but evidently lost out. It would stand a chance for 6C, however, depending on market pressure.

29. FIP Priorities (?)

Question. Would it be possible for the priority of heavy disk users to be dynamically adjusted in such a way that overall system performance isn't degraded by one or two users?

Discussion. The RSTS/E File Processor (FIP) currently works on a simple first-in, first-out basis. This means that a job having light CPU requirements but using much disk I/O can slow the system down significantly, particularly when SY: is also used for swapping. Unfortunately, using $PRIOR (or the automatic priority balancer either) to adjust priority has little effect when a job doesn't need much CPU time. What's needed is for FIP to help balance how many disk accesses each job gets.

30. Maximum Filesize 65535 (?)

Question. The maximum size of a RSTS/E file is 65535 512-byte blocks. Many applications require a larger total amount of data; is there to be any help?

Discussion. This question was posed at the DECUS in Los Angeles. The response from DEC at that time was that enlarging the file maximum would be impractical, and it went out in the last Channel 13 as an Answer. The furor isn't over, however; judging from the unsolicited response at Atlanta, this one should remain a newlsetter item.

31. Clustersize Multiplier (?)

Question. For really large files, we'd like to get around the present 256 ceiling on clustersize.

Discussion. One possibility: could we use a "multiplier" to extend the range of the existing four-bit field? A clustersize of "4" could mean four times the pack clustersize, for instance.

32. Open by Filename (?)

Question. Basic-Plus requires that all references to each file be by channel number. It can get difficult to allocate channel numbers ahead of time, particularly when a program is to be made up to include standard routines from an APPEND library.

Discussion. One possibility would be to let OPEN simply use the next available channel number, and to return it for later use in the program.

33. Ordered REORDR (?)

Question. It's a peculiarity of RSTS' file handling that the files most recently written are precisely those most difficult to get to. Directory entries are made in chronological order, and lookups are normally made starting at the front, or oldest entry. This can mean a long directory search for commonly-used files. For example, many (if not most) installations modify $LOGIN, so $LOGIN.BAC is often near the end of [1,2] — a bad situation for such a frequently-run program. A similar situation can easily occur for application files.

Discussion. Several possibilities for improvement exist. Directories could be inverted: that is, directory searching could be made to start with the most recent entry. Better would be to have REORDR sort directory entries so that the user could specify which should be the easiest to retrieve. Probably better still would be to keep access counts in MFD's and UFD's. REORDR could then bubble to the front those files actually used most frequently over some specifiable recent period.

34. Privilege-ing Sys Calls (?)

Question. In a hostile environment, the availability of certain sys calls may prove tempting to young, inquisitive minds. To avoid providing the temptation, and risking compromise of system security, would it be possible to selectively make sys calls illegal for any but privileged users?

Discussion. This has already been enough of a problem in the field to cause patches to be written for a couple of the most sensitive cases. They're simple, one-word changes, and are described in Software Support's weekly software notes. At the moment, nothing on the order of, say, a Syssen option has been planned.

35. Non-Privileged RECEIVE (?)

Question. A program must now be privileged to be allowed to declare itself a message receiver. This deprives a shop of writing detached utility programs which could be accessed by general users via SEND-RECEIVE.

Discussion. Evidently one of the main reasons form making RECEIVE privileged was to keep a naive user from knotting up a large number of small buffers. The kind of facility requested here could be simple-minded enough to avoid that risk. It could take the form of SEND + wait for response, with perhaps a system-defined limit on the returned data; the entry in the RECEIVE tables would exist only during the wait.

36. RECEIVE With Timed Sleep (!)

Question. Currently, a program desiring to check for pending messages must use one of two options. It can attempt to simply RECEIVE, in which case an error will indicate no messages pending. Or, it can RECEIVE with Sleep, and be awakened only when there is a message to process. There are cases in which the receiving program has other things to do besides those dictated by incoming messages. It would be a good idea to allow specification of a maximum SLEEP time in RECEIVE with Sleep.

Discussion. Agreed. RECEIVE with timed Sleep will be available in version 6B.

37. Half-Duplex Support (!)

Question. Many users would like to make use of half-duplex facilities.

Discussion. This has been planned; originally scheduled for version 6B, it's now slipped to 6C.

38. Remote LP: (?)

Question. We need support for remote line printers, run via serial lines. Some users have, of course, written B+ programs to handle forms alignment, etc on remote KB:'s, more or less like printers, but more is needed.

Discussion. There are actually several questions involved here. One, remote line printers raise issues about spooling protocol: RJE sites vs terminal associated printers, etc. Different printers would need to be treated in different ways by the spoolers. The QUEMAN-SPOOL section of the Cusp's is currently being re-designed, and many considerations are for the moment open questions. Incidentally, any suggestions you may have in this area are more than welcome, and will be faithfully forwarded to Maynard. Two, effective channel use may suggest half-duplex for many remote line printers. The issue of RSTS/E support for half-duplex is likewise still open for discussion. Third, we might want to drive these remote printers directly (software-wise, avoiding the overhead of the TTY handler). This raises conflicts over how to treat modem lines, which currently must always be assumed to be KB:'s. (RJ2780 ignored.) It seems doubtful if we'll see these questions answered in time for any kind of implementation in version 6B. Finally, we might mention that at least one OEM offers a set-up consisting of a microprocessor, a half-duplex line, and another microprocessor to hook up a remote LP:. The microcprocessors can handle line turnaround, communicate forms-out by Control-S (for example), and so on.

39. VT55 Graphics (?)

Question. What about graphics support for the VT55 scope?

Discussion. There are plans to support VT55 graphics throush a product to be developed by the Laboratory systems group, but there is no official schedule.

40. LSI-11's for Terminal Handling (!)

Question. Are there plans for special terminal-handling gear to be offered incorporating the LSI-11 microcomputer?

Discussion. The use of a separate front-end processor CPU is not obvious for RSTS/E. However, it is predictable that future hardware, both CPU's themselves and peripheral controllers, will increasingly employ microprocessors.

41. Extra Disk Cylinders (!)

Question. Most disk media have extra cylinders to compensate for possible bad blocks. Does DEC plan to make use of them in RSTS?

Discussion. There are no present plans to do so.

42. BATCH Error Recovery (?)

Question. BATCH needs a facility for conditional execution of job stream segments, and for detecting when errors have occurred in the controlled jobs.

Discussion. There are to be some minor enhancements in this area for version 6B: system error messages will be prefixed by a "?" for "fatal", "%" for "warning". It will be possible to instruct BATCH to cancel or continue the job at any given level. The larger question remains, however — in the coming rewrite of the QUEMAN-BATCH-SPOOL system, what kind of facilities do we really need?

43. Switching PPN's Within a BATCH Stream (!)

Question. It is sometimes desirable to coordinate execution of work to be done under more than one PPN under BATCH.

Discussion. Hooray! A facility for doing this in fact exists. It is possible for a privileged user to queue for Batch processing a stream containing $JOB cards which specify other PPN's. There can be more than one $JOB card in a single Batch stream, and they need not have the same PPN.

44. Further Batch Suggestions (!)

Question. Pursuing the idea of a BATCH re-worked to our specifications, here are some other ideas which have been suggested. Imbedded batch streams - that is, the ability to specify an "indirect command file" to BATCH. Variable symbols - the ability to create a batch stream having automatically patchable entries, such as file names, which might be changed each time queued.

Discussion. We won't ask for a vote on specifics, but rather solicit further suggestions. See the other discussions on QUEMAN/BATCH/SPOOL - it's Them asking Us this time.

45. LOGIN Speed Setting (?)

Question. It would be helpful if $LOGIN could recognize at least the difference between the two most common dial-up speeds: 110 and 300 baud.

Discussion. It does complicate operations to have to maintain two sets of dial-up lines simply to provide dirfferent speeds. There is already in the field at least one patch (circulated throush this newsletter) which does a nice job of distinguishing 110 and 300 baud, at least on DC11 & DH11 lines. It's possible already - but more users would benefit if it were a standard feature of DEC software.

46. LOGIN by Mnemonic (?)

Question. A facility to respond with an alpha name to LOGIN's prompt for a PPN.

Discussion. In a shop where a clerical or operations staff run jobs for many users, the standard PPN structure can become somewhat difficult to use. The problem of too many passwords can be ameliorated by setting aside certain PPN's for this kind of batch use. They can all then be given a common password, which can be changed frequently for security. PPN's themselves are another matter; the operations staff must accurately use a larger set of PPN's than does any normal timesharing user. Often, too, the programs being used are from the library, so a wrong PPN simply results in an erroneous charge. What can help is to allow $LOGIN to accept a mnemonic name which it can translate into a PPN.

47. CUSP to Free up Empty UFD's (?)

Question. "I would like to have a CUSP which would free all blocks in empty UFD's. Accounting data for the deleted UFD should not be altered, nor should the order in which this account appears in the MFD."

48. Cross-Reference & RESEQ (?)

Question. Neither CREF nor RESEQ has ever been an officially-supported product. With the coming of extended Basic-Plus, will there be successors for these programs?

Discussion. Obviously, extended variable names (if nothing else) implies major changes in any program working with Basic-Plus syntax. Both of these programs are often used in any shop having substantial programming activity, whether developmental or maintenance.

49. Pure-Code RUNOFF (?)

Question. There have been rumors that there exists a RUNOFF written in MACRO. Some of us use RUNOFF quite a bit, and a faster version would indeed be welcome.

Discussion. Rumor confirmed: there is a RUNOFF written as a run-time system, and it's consequently much faster than the B+ version. It currently exists only in a form for internal use, and also uses a somewhat different command syntax. It is possible, however, that it might be developed into a product.

50. SORT Commentary (!)

Question. What can be said about the various versions of SORT for RSTS/E? Some of us are confused.

Discussion. There are currently two DEC sorts for RSTS/E, and several on the market and/or coming from OEM's. The first DEC sort is written in Basic-Plus, and was originally offered as part of the Commercial Extensions package. The second (called SORT-11) is implemented via a run-time system, and although originally offered as part of the COBOL package, is now available alone as well. We asked if the Basic-Plus sort was to be rewritten: the answer seems to be No, with the implication that future B+ offerings will be written to make use of RMS-11. And what about SORT-11? Users and DEC alike feel there's room for performance improvement. Plans here call for 1) an immediate effort at soupins-up, but not linked to 6B; and 2) an eventual complete rewrite, probably linked to RMS-11.

51. User Password Changer (?)

Question. A simple utility to allow a non-privileged user to change his own password.

Discussion. Several installations provide this facility to their timesharing customers. Making the user responsible for his own password security makes it possible for him to arrive at his own tradeoff between frequent changes and possible piracy. It relieves system management of a thankless job, too.

52. Who-Is Utility (?)

Question. It would be useful for a timesharing user to be able to ask the system quick, simple questions about system status.

Discussion. What's meant here is partially a streamlining of $SYSTAT, and partially some extensions. For instance, typing "WIZ 15,14" or "WIZ [15,14]" might obtain the keyboards logged in under [15,14] together with a descriptive name for that PPN. (This could come from the system's accounting data, or from a user area chained to the MFD, mentioned elsewhere in this wish list). Similarly, "WIZ DP1:" could give the ID of the pack mounted on DP1:. "WIZ LP3:" might show the current forms mounted on LP3:, together with the name of whatever user might have the device in use or ASSIGN'ed.

53. FILEX RT11-RSTS/E File Exchange Program (!)

Question. How about a utility file exchange program to enable communication between the RT11 and RSTS/E file structures?

Discussion. There is none planned for version 6B, but it's likely to be offered later.

54. PIP De-assign (?)

Question. If one is doing, say, PIP to a line printer, and the job is interrupted by a control-C, the line printer remains tied up although printing stops. Couldn't this be fixed?

Discussion. Control-C is an asynchronous program-stopper, but it doesn't cause files to be closed, execution to go through END, etc. We wouldn't want it to, either - it's frequently desirable to CONTINUE after doing a control-C and some immediate-mode snooping. On the other hand, a naive user is wanting only to "stop everything" when he control-C's a PIP, and is confused that the program remains live, in core, and that its devices aren't released. PIP could easily be improved by having it trap control-C's, and goto END when one is typed.

55. DEC Adoption of User Programs (!)

Question. There are many good CUSP's out in the world, written by users. It has been suggested on more than one occasion that DEC give thought to taking the best of them, revise them to standards, and distribute them under maintenance.

Discussion. In Atlanta, a show of hands indicated that a substantial majority of users present had either modified DEC CUSP's or written CUSP's of their own. RSTS was constructed with the intent of making the system maintainable by having much of system service implemented in Basic-Plus. User-written CUSP's undoubtedly improve RSTS' effectiveness in individual instances, and undoubtedly too, some of them would be helpful to many other users as well. However, consider that centralized maintenance must be the first criterion for manufacturer-supplied software. To make a typical CUSP fit well with the system as a whole, and to make its enhancements general enough to appeal to a wide market, it would usually be necessary to rewrite, not just revise. This is not to say that user input isn't valuable; quite the contrary. It's just that the useful input is not code, but functional specifications, writeups of what is done, and how. This is really what we should want to have a say on, and DEC is not only willing to listen to functional requirements, they've explicitly asked for our help. The CUSP's are going to be re-written, every one; now is the hot time to make suggestions that will actually show up in supported software. (Version 6C, by the way; not 6B.) If you have, or can get, anything in writing, send it in. Where CUSP's are concerned, it's less of a question of Wishlist than of providing guidelines for an effort that's already committed.

56. MACRO under RSTS/E (!)

Question. Will use of assembler language (MACRO) be supported under RSTS/E?

Discussion. There are no such plans beyond the use of MACRO subroutines to support Fortran programs. MACRO use in this context is somewhat limited, in that it supports neither I/O nor monitor calls (EMT's).

57. CALLs Across Languages (?)

Question. What about a program written in, say, Basic-Plus, which would like to CALL a subroutine written in, say, COBOL?

Discussion. This is a complex issue, currently under consideration.

58. Chaining Across Languages (!)

Question. Will it eventually be possible to chain from a program written in one language (say Basic-Plus) to a program written in another language (say Fortran)?

Discussion. This is a question involving a certain amount of monitor support, since checking for presence of the appropriate run time system, etc will be required. It is likely to be provided (although it will likely be later than version 6B).

59. All-device Support (!)

Question. Some languages (eg, Fortran) support only a subset of the available RSTS/E peripheral devices. Is there any chance that version 6B can offer support of all devices by all languages?

Discussion. No.

60. Fortran SYS calls (?)

Question. Could Fortran be given access to the monitor services provided to Basic-Plus by SYS calls?

Discussion. One possible implementation could be by offering a set of library routines which could be CALL'ed.

61. String Arithmetic (!)

Question. Version 6B is touted as offering string arithmetic, and there's some rumble of necessary tradeoffs. Explain.

Discussion. String arithmetic is a facility to do arithmetic on numbers represented in core as ASCII strings (including possible decimal points and leading signs). It's done by software routines (not directly by hardware), and there's no essential limitation on precision, since the "number strings" can be of indefinite length. Not surprisingly, string arithmetic is several times slower than native-mode computation - but this often doesn't matter much, especially in commercial applications. Now, about the tradeoffs. In every RSTS/E sysgen we choose various options to be included in our run-time system.

Among them are trigonometric functions, matrix manipulation, and PRINT-USING. String Arithmetic will be another in this list of options - but now, for the first time, we're in RTS core trouble. The entire list of options adds up to more storage than there is available for the run-time system address space, so it won't be possible to build in the whole list of them. We used to be able to simply build in everything, but there will be so many goodies in 6B that some selectivity will be mandatory.

62. Execution of Strings (?)

Question. For certain applications, particularly CAI, it would be useful to be able to pass a string to the B+ parser for immediate-mode execution.

Discussion. This would essentially be taking advantage of existing facilities in a new way - that is, using immediate mode under control of a running program. For instance, a CAI program asks a student for a numeric answer. The student doesn't know the answer, but does know how it would be computed in a short line. So the student types in a short line; the program executes the string, prints it out, and compares it with the correct answer. To do this now would essentially require writing a statment parser in Basic-Plus, a nasty job. (Footnote: this technique would be useful only with Basic-Plus "One", since the compiler-type Basic-Plus Two won't have run-time expression evaluation.)

63. Function Library (?)

Question. How about installation-defined and user-defined function libraries, to extend the set of facilities available through the system functions?

Discussion. Other languages frequently offer such things, especially through the use of subroutine libraries. Function libraries would be easier to use, however, since "A$=fnQ$(X%)" is all one would have to write, and no linking would be involved. It would be especially nice if frequently-used installation-standard functions could be coded in MACRO.

64. Changing Function Arguments (?)

Question. It's sometimes necessary for a routine to be passed several arguments, some of which are to be altered upon returning. Could functions be allowed to change their arguments?

Discussion. This question is related to the issue of local variables for functions and subroutines, which along with many others, will get new scope in Basic-Plus Two. Currently, we must set dummy arguments and do a GOSUB, then retrieve the arguments. Alternatively, a series of functions can be used if the situation isn't too complex.

65. WHILE-NEXT (!)

Question. If you've read some of your CUSP's carefully, you've come across a construction like:

```
WHILE A$ <> ""
   blah
   blah
   A$ = RIGHT(A$,2%)
   blah
NEXT
```

It's scary (not documented), but handy. What's the story?

Discussion. There are "clever" programmers in every installation. The WHILE-NEXT construct was an accidental feature in the way Basic-Plus is implemented. It got discovered, then used in CUSP's, then used in the field. Because it's turned out to be useful, WHILE-NEXT will be officially included in Basic-Plus-Two, and will explicitly not be eliminated from "plain old" B+. A word of warning is in order, however, for "plain old" B+. Since WHILE-NEXT was accidental, B+ doesn't check it for validity, and it's possible to get in trouble using it improperly. In particular, exiting sideways (not falling out through the NEXT) may leave the stack dirty and cause a crash.

66. RESTORE 560 (?)

Question. Presently, there is no way to go directly to a particular part of a program's DATA statement pool. Syntax of the form "RESTORE <line number>" would be a significant extension.

Discussion. The best current alternative seems to be inserting some "X" into the DATA pool and then to: RESTORE READ A$ UNTIL A$="X". This has the same effect, but is obviously slower.

67. DECLARE or TYPE (?)

Question. Basic-Plus uses variable-name suffixes to declare type (the % for integer, $ for string, null for real). In version 6B Basic-Plus, variable names can be up to 29 characters. The final (30th) character is still to be a type designator (the familiar % or $). Other languages (eg, Fortran) use different schemes, which can provide for a verb like DECLARE, which allows any variable name to assume any type. How about some RSTS facility for allowing type of variable to be explicitly declared ?

68. PRINT Abbreviation (?)

Question. Don't let use of the ampersand for "PRINT" die!

Discussion. (The ampersand can presently be used as an abbreviation for PRINT. In version 6B, it will be used to signal line continuation.) Obviously, PRINT is an often-used verb, and program listings get knotted up around clusters of PRINT statements. Immediate-mode statements are special beneficiaries. Although some other character might do as well, having an abbreviation is handy. We could let line continuation be signalled by ampersand-carriage return, and have ampersand-anything else mean "PRINT".

69. LIST Flagged Statements (?)

Question. A command like LIST ?100-400 to list all questionable statements between lines 100 and 400.

Discussion. Presently, when using a CRT, one must OLD the program, copy the line numbers displayed for faulty statements, and then type LIST followed by the line numbers. This is the kind of job a computer can easily do better than us humans.

70. Partial SAVE's (?)

Question. A command like SAVE LP:/100-400 to get a partial listing or SAVE SY:/50-90,210-260 to construct the nucleus of a new program.

Discussion. The only way to do this kind of thing presently is to OLD the whole program, DELETE the unwanted parts, and then SAVE the residue. The keyboard command LIST does a good job, but works only at the (often slow) user terminal.

71. EXTEND'ed Basic-Plus Filter Program (!)

Question. By now we've all heard about Basic-Plus-Two, and how a translator program will be provided for the switchover. Could a similar program be provided to help make the (smaller) change from current Basic-Plus to Basic-Plus Extended?

Discussion. If all this begins to make it sound like there are three versions of B+ in the works, it's because it's essentially true. First there's plain old B+, which we're using now. Then there's B+ Extended, which is plain old B+ with the ability to handle long variable names, indefinite-length lines, imbedded comments, etc. Thirdly, there's Basic-Plus-Two, which will be a true compiler-type language, will offer CALL's, overlays, RMS-11 support, etc. The latter is to be a separate product, at an extra price, and available shortly after version 6B comes out. The first two are really two faces of the 6B version of "plain old" B+. "EXTEND" - "NO EXTEND" commands are to be provided so that Basic Plus will be able to continue to process our currently-running programs without change. Many of us, however, will probably want to make the Extend'ed format our installation standard. Eventually we'll have to face the question of converting current software to the Extend'ed syntax. Since this conversion should look a lot like a subset of the conversion from "plain old" B+ to Basic-Plus-Two, it should be possible to make a relative of the B+ II translator do the job. DEC agrees that it's possible, and will try to provide such a translator, time permitting.

72. Variable-Name Expander (?)

Question. With the coming of long variable names, many installations will want to begin improving documentation of their program base.

Discussion. A useful one-time conversion utility would be a program which could scan a Basic-Plus program, replacing selected variable names by expanded names from a user table.

73. Basic-Plus-Two Debugging (!)

Question. Is there any information on what kind of debugging facilities are going to be offered with the initial version of Basic-Plus Two?

Discussion. Nothing official has been announced; however, this area has received considerable attention both within DEC and in this newsletter, and is likely to be a critical factor in users' deciding whether to upgrade. The unofficial rumors at Atlanta went like this. Debugging will be possible without re-compiling. Breakpoints can be set, perhaps by a syntax like "BREAK 9350", issued before "RUN". At the breakpoint, other breakpoints can be set/cleared, and execution resumed. It will be possible to "STEP", which would execute one numbered line. Printing of scalar variables will be permitted; so will single array elements. No variables or expressions will be allowed in subscripts, however; statement modifiers will be illegal, too. (Remember, we are talking about the object-time debugging language here; these are not restrictions in Basic-Plus.) Variables: new values can be assigned to both scalars and individual array elements. The values can be literal values; the same restrictions about no expressions and no modifiers apply to debug assignment commands.

74. RMS-11 (!)

Question. There's been much discussion of RMS-11 (RMS = Record Management System). What's going to be available in version 6B; what about Basic-Plus versus Basic-Plus-Two?

Discussion. RMS-11 will provide logical record handling: access, deblocking, etc. It is not a "database" manager. The hooks for two of its facilities will be built into 6B — sequential and relative I/O. The third facility — indexed I/O — will be available only as an extra-cost, separate product. Use of all three facilities will be initially possible only through Basic-Plus-Two, itself an optional, extra-cost product. There are no plans for making RMS-11 accessible through "old" Basic-Plus at all.

75. 4MB Support for 11/70 (!)

Question. Support for the PDP-11/70's full memory capacity?

Discussion. Yes; in version 6B.

76. RJ2780 Space Compression (!)

Question. Will a space compression feature be added to RJ2780?

Discussion. There are no current plans to do so.

77. Run-Only System (?)

Question. Some installations could use the ability to construct a run-only system.

Discussion. Some machines run compiled programs only, and are not used for development work. It would be possible to save on operating system core requirements by making the Basic-Plus source interpreter an optional part of the system.

This new RSTS user profile was developed by Elizabeth Matlack of Dow Chenical and Howard Kipple of New Mexico Junior College. Please send your completed profile to:

BILL TABOR RSTS SIG
% DECUS OFFICE
146 MAIN STREET
MAYNARD, MASS. 01754

RSTS USER PROFILE

Date of Installation: _____

Contact Person _____

Address: _____

City, State, Zip _____

Company _____ Phone _____

For multiple installations please reporduce this form and respond for each installation.

Respond with quantity of each item.

Processor: 11/20 _____ 11/40 _____ 11/45 _____ 11/50 _____ 11/70 _____ Other _____

Operating System: V4A ___ V5A _____ V5B _____ V5C _____ V6a _____ V6B ___ Other _____

Languages: COBOL _____ Basic + _____ Fortran _____ Dibol _____ Other _____

Memory Size: MOS _____ Cache _____ Core _____ Bipole _____ Other _____
(WORDS)

Vendor of other _____.

RJE 2780 Corrected to: 360 ___ 370 _____ Other _____

Card Reader: CR11 _____ CM _____ Other _____

Card Punch: _____

Line Printer: LS 11 _____ LP 11 _____ Other _____

Terminals: TTY _____ LA 30 _____ LA 36 _____ Other _____

Vendor _____.

CRTS: VT 05 _____ VT 50 _____ VT 52 _____ Other _____ Vendor _____

Lines: DH 11 _____ DN 11 _____ DL 11/A _____ DL 11/E _____ Other _____

Disk Drives: RK 02 _____ RK 05 _____ RP 02 _____ RP 03 _____ RP 04 _____

RS _____ RF _____ Other _____ Vendor _____

Mag Tape: 7 Track _____ TU 10 _____ TU 16 _____ 9 Track _____

Other _____ Vendor _____

DECTAPE: TU 56 _____

Floppy Disk _____

Current Applications: _____

_____

Need Software For: _____

_____

_____

FIRST CLASS
PERMIT NO. 33
MAYNARD, MASS.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

Postage will be paid by:

DECUS

DIGITAL EQUIPMENT COMPUTER USERS SOCIETY
MAYNARD, MASSACHUSETTS 01754